Saudi Journal of Engineering and Technology (SJEAT)
Scholars Middle East Publishers
Dubai, United Arab Emirates
Website: http://scholarsmepub.com/

ISSN 2415-6272 (Print)
ISSN 2415-6264 (Online)

Privacy Preservation using LDSS CP-ABE Scheme for Mobile Cloud Computing

Anusha R[1*], Dr. S Meenakshi Sundaram[2], Asha P[1], Bhargavi Y[1], Bindu Jayaram[1]
[1]Department of CSE, GSSS Institute of Engineering and Technology for Women, Mysuru, India
[2]Professor and Head, Department of CSE, GSSS Institute of Engineering and Technology for Women, Mysuru, India

Original Research Article

***Corresponding author**
Anusha R

Article History
Received: 12.06.2018
Accepted: 22.06.2018
Published: 30.06.2018

DOI:
10.21276/sjeat.2018.3.6.7

Abstract: With computer operations and mobile device technologies expanding, mobile cloud computing has been shaping up as the future of web-based communications. Data security has consistently been a major issue which restricts the further development in mobile cloud. Many measures have been undertaken to improve in cloud computing. As the mobile devices have finite resource and power most of the studies on cloud computing are not convenient on mobile devices. Hence it becomes requisite to diminish the computational overhead in mobile cloud computing. In this paper we come up with a scheme called as light weight data secure sharing scheme for mobile cloud. It embraces CP-ABE which is one of the most suitable technologies for data access control in cloud storage system but alters the framework of access control tree to make it appropriate for mobile cloud environments. External proxy servers are being introduced to reduce the computational overhead on mobile device. Attribute description fields are used to implement lazy revocation which depletes the user revocation cost. The investigational outcome unveils that LDSS is capable to decrease the overhead issues on the mobile device when the data is being shared in mobile cloud environment.

Keywords: Web-based communication, Cloud computing, Mobile cloud environment, Computational overhead.

INTRODUCTION

To a great extent many cloud based mobile applications are being used. Various documents such as photos, videos and other files are uploaded to the cloud by the data owners. These data owner share their data with the other data users whom they opt to share. Since the confidential data files are being uploaded, data privacy of these data files is a huge hassle for data uploaders.

The data retrieval service provided by the cloud service provider is neither adequate nor beneficial. They cannot meet the prerequisite of the data owner. Initially, when data files are uploaded onto the cloud by data owners, the data files are not under the supervision of the data owner and the cloud service provider may mismanage the data files for profitable concern and other various reasons.

Subsequently, data owner whom they select to share the encrypted data it becomes obligatory to consign password to the users, which becomes inconvenient. Data owner can split data users into many clusters and send the passwords to these clusters. To rationalise the access privilege management fine grained access control is required for this approach. For these two cases password handling is a huge problem. Confidential data must be encrypted before being uploaded onto the cloud service provider, to solve the above problem. Yet data encryption comes up with some new obstacles.

Indulging methodical access control mechanism on ciphertext decryption so that only the sanctioned users can make use of decrypted data is challenging. Further data owner can concede or revoke data access privileges only if they are provided with the effective user privilege management proficiency. In this paper, the following hypothesis are considered,

- Cloud Service Provider is regarded to be genuine and curious.
- Before being uploaded onto the cloud, the data files must be encrypted.
- Permission to the data users to retrieve the data file is achieved through encryption and decryption key allocation

The present approaches are suitable for non-mobile cloud environment, as they consume huge storage

and resources, which is not convenient for mobile devices. The main objectives of the proposed scheme are:

- To extend an efficient access control over ciphertext, we propose al algorithm called LDSS CP-ABE based on the attribute-based encryption.
- To perform an encryption and decryption operation we use proxy servers.
- In order to deal with user revocation problem Lazy re-encryption fields are proposed.
- Lastly, light weight data secure sharing scheme framework on data sharing is implemented. Experimentation on LDSS depicts significant decrease of the overhead on client side and only minimal additional cost on server side is introduced.

RELATED WORK

A. Sahai And B. Waters *et al.*, [1] have introduced a new type of Identity-Based Encryption (IBE) scheme that they called Fuzzy Identity-Based Encryption. In Fuzzy IBE they viewed an identity as set of descriptive attributes. V. Goyal, O. Pandey, A. Sahai and B. Waters *et al.*, [2] have introduced Attribute-based encryption for fine-grained access control of encrypted data. As more sensitive data is shared and stored by third-party sites on the Internet, there will be a need to encrypt data stored at these sites.

R. Ostrovsky, A. Sahai and B. Waters *et al.*, [3] introduces Attribute-based encryption with non-monotonic access structures. They construct an Attribute-Based Encryption (ABE) scheme that allows a user's private key to be expressed in terms of any access formula over attributes. B. Waters [4] introduces Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. He presented a new methodology for realizing Ciphertext-Policy Attribute Encryption (CP-ABE) under concrete and noninteractive cryptographic assumptions in the standard model.A.B. Lewko, T. Okamoto, A. Sahai, K. Takashaima and B. Waters *et al.*, [5] introduces fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. They presented two fully secure functional encryption schemes: a fully secure attribute-based encryption (ABE) scheme and a fully secure (attribute-hiding) predicate encryption (PE) scheme for inner-product predicates. Qihua Wang, Hongxia Jin *et al.*, [6] introduces Data leakage mitigation for discretionary access control in collaboration clouds. With the growing popularity of cloud computing, more and more enterprises are migrating their collaboration platforms from in-enterprise systems to Software as a Service (SaaS) applications. Wang W, Li Z, Owens R *et al.*, [7] introduces Secure and efficient access to outsourced data. Providing secure and efficient access to large scale outsourced data is an important component of cloud computing.

PROPOSED WORK

We present a Light Weight Data Sharing Scheme for Mobile cloud. The six main modules of the LDSS are as follows:

1. Data Owner(DO):
   A Data Owner transfers the data file to the mobile cloud and share it with the authorised user. The polices required foe the access control of the data files is governed by the data owner.
2. Data User(DU):
   The uploaded data files in the cloud is fetched by the authorised users.
3. Trusted Authority(TA):
   The generation and the distribution of the keys is supervised by the trusted authority.
4. Encryption Service Provider(ESP):
   It provide services to data owner to encrypt their data files before being uploaded to the cloud.
5. Decryption Service Provider(DSP):
   For the decryption of the data files by the user. This service is provided by the DSP.
6. Cloud Service Provider(CSP):
   The uploaded data files by the data owner are stored in the cloud.

As shown in the Fig 1, below on receiving the public key from the trusted authority the data owner uploads the file to the cloud. Since the cloud is not completely trustworthy it becomes requisite to encrypt the data files. At this scenario ESP comes into picture, which provides services to encrypt the files. The data owner specify the policies for the access of the data files in the form of access control tree. For the genuine data user it becomes vital to satisfy the policy to access certain data files. Symmetric encryption mechanism are used for the encryption all data files. ABE is exercised to encrypt the generated symmetric key. The specified policies for the access of the data files are submerged in the ciphertext of the symmetric key.

On obtaining the attribute key from the trusted authority and satisfying the policies of the access control on data files the data user will be capable to decrypt the ciphertext and fetch the symmetric key. The mobile users experience exhaustive data processing of the encryption and decryption services, in order to mitigate this exhaustive data processing ESP and DSP are introduced at both side of the mobile devices.

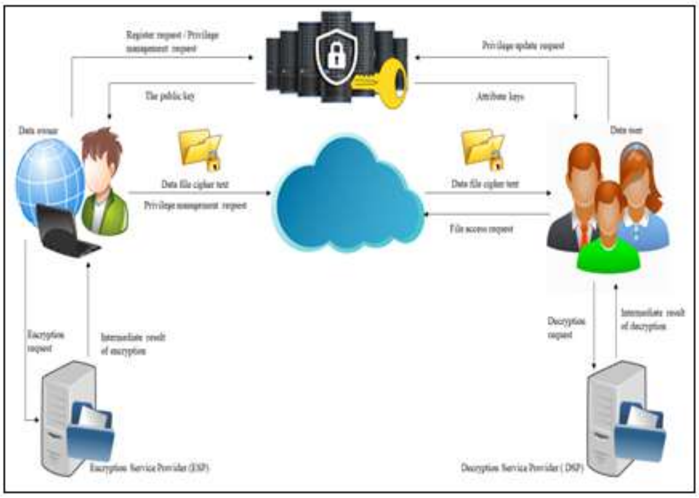

Register request / Privilege management request
The public key
Privilege update request
Attribute keys
Data owner
Data user
Data file cipher text
Privilege management request
Data file cipher text
File access request
Encryption request
Intermediate result of encryption
Decryption request
Intermediate result of decryption
Encryption Service Provider (ESP)
Decryption Service Provider (DSP)

Fig-1: System architecture

METHODOLOGY ADOPTED

LDSS-CP-ABE algorithm is designed using the following functions.

Setup (*A*, *V*): Generate the master key *MK*, the public key *PK* based on attribute set *A* of the Data Owner and the version attribute *V*.

KeyGen (*Au*, *MK*): Generate attribute keys *SK*u for a data user *U* based on his attribute set *A*u and the master key *MK*.

Encryption (*K*, *PK*, *T*): Generate the ciphertext *CT* based on the symmetric key *K*, public key *PK* and access control tree *T*.

Decryption (*CT*,*T*,*SK*u): Decrypt the ciphertext *CT* using the access control tree *T* and the attribute keys *SK*u .

Function Setup () is called by the trusted third party (TA) to generate the master key and the public key. The master key is used to generate attribute keys and the public key is used to encrypt data files.

Function 1: Setup ()

INPUT: The attribute set A, the version attribute V.

OUTPUT: The master key MK, the public key PK.

- Construct a p-order bilinear group G0 of generator g and a bilinear mapping e: $G_0 * G_0 = G_1$.
- Randomly choose a,b□Z_p and calculate g^b ,e(g, g)^a.
- For each attribute ai in A, randomly choose t_i □ Z_p, and calculate $X_i = g^{ti}$
- For V, randomly choose t_v□ Z_p , and calculate X $_v$=g^{tv} .
- Return the master key MK and the public key PK, Wherein MK={a,b}, PK={ G0 , g , g^b, e(g,g)^a, {X_i }$_{i=1}^{k}$**, X_v}.**

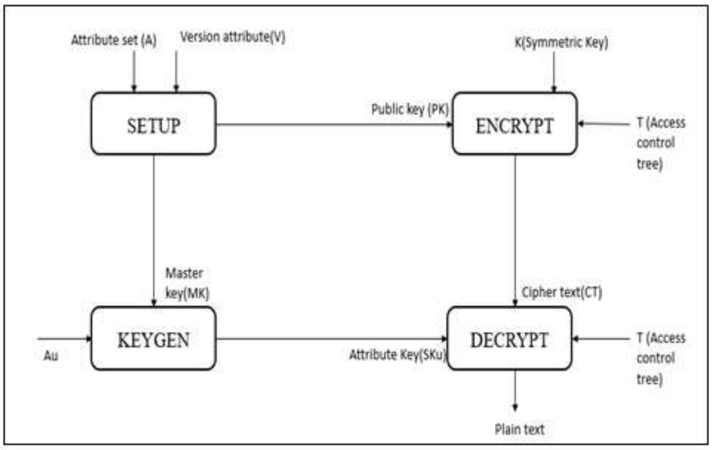


Fig-2: LDSS CP-ABE fundamental algorithm

Function 2: KeyGen()

INPUT: The attribute set *A*u, the master key *MK*={*a*,*b*}.

OUTPUT: Attribute keys associated with *A*u.

- Randomly choose a parameter, $r\square Z_{P}$, and calculate $SK_r=g^{(a+r)/b}$
- For each attribute a_i in A_u, randomly choose , and r_i $\square Z_p$, and calculate $SK_r = \{g^{ri} ,gr . X_i{}^{ri} \}^{j}_{i=1}$
- For V, randomly choose r_v $\square Z_p$, and calculate $SK_u = \{ g^{rv}, g^{r} . X_{v}{}^{rv}\}$
- Return $SK_u = \{SK_r, Sk_a, SK_u\}$

Function 3: Encryption()

INPUT: The symmetric key *K*, public key *PK*, access control tree *T* (including the left subtree *T*a, right subtree *T*v, and left subtree has *num* leaf nodes).

OUTPUT: The ciphertext *CT*.

- Randomly choose S $\square$ Z_p as the secret of *T*, and calculate $CT_k=\{g^{bS}, K\, e(g,g)^{aS} \}$.
- Get the value of the two children (namely S_a, S_v) of the root node according to the access control tree.
- Calculate $CT_v=\{ g^{s}{}_{v\,,}\, g^{r} . X_v{}^{s}{}_{v} \}$
- Return CT={ CT_k, CT_a, CT_v }.

Function 4: Decryption()

INPUT: Ciphertext *CT*, the access control tree *T* (including the left subtree T_a, right subtree T_v, and left subtree has *num* leaf nodes), SK_u (attribute keys of user *U*).

OUTPUT: The plaintext of *K*.

- Randomly choose t, and get SK_u'={ SK_r'= $SK_r{}^{t}$, SK_a, SK_v }.
- For every leaf node *z* of T_a, calculate DecryptLeaf(CT_a, SK_u', *z*) = $e(g , g)^{qz(0)}$.
- For the leaf node in right subtree, calculate DecryptLeaf(CT_v, SK_u', *V*) = $e(g , g)^{qv(0)}$.
- Let CT_k-1= g^{bS}, CT_k-2= K $e(g, g)^{aS}$, calculate *K*

RESULTS & DISCUSSION

The experimental results show that LDSS can ensure data privacy in mobile cloud and reduce the overhead on users' side in mobile cloud. Figures 3 to 13 shows work flow of the LDSS. External proxy servers are introduced to reduce computational overhead on client side.

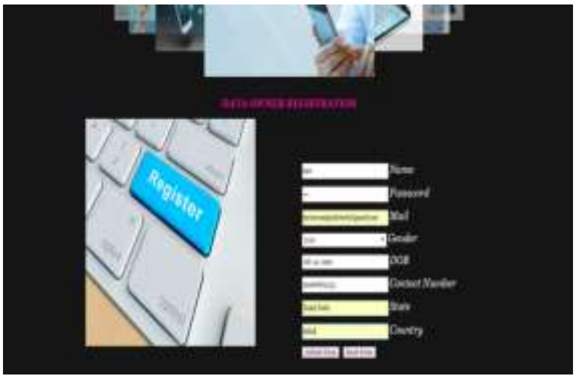


Fig-3: Data owner registration form

Figure-3 depicts the data owner registration form containing his/her personal information and the password with the same user name and password he can login

Fig-4: Public key request

Figure-4 depicts the public key request to trusted authority from data owner. Figure-5 depicts generation of public key by TA which is sent to the DO mail ID. Figure-6 depicts the uploading of files onto the cloud by using public key and specifying the access policies. Figure-7 depicts the data user registration form same as data owner

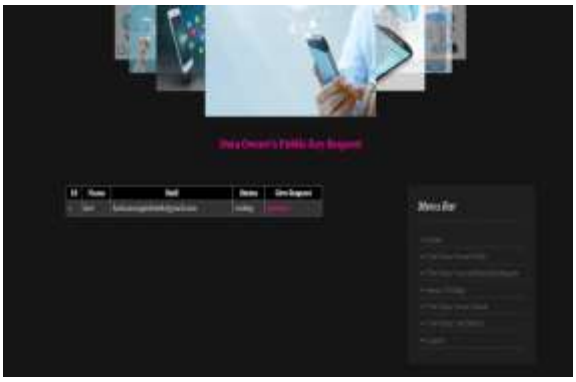


Fig-5: Granting the public key request

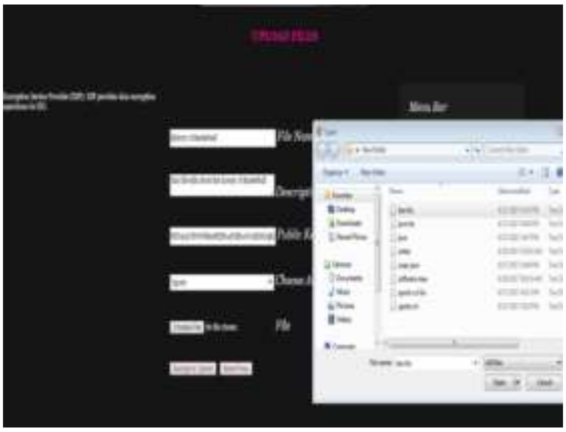
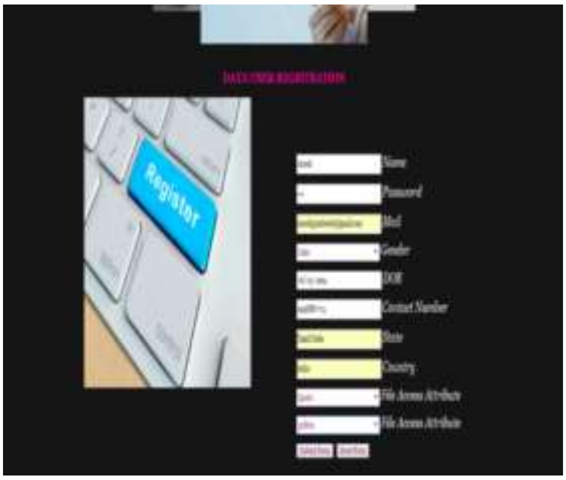
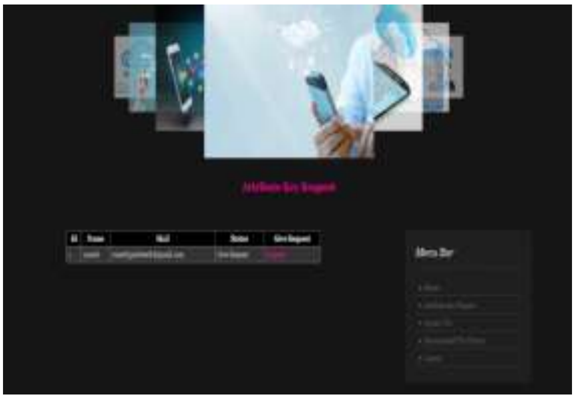


Fig-6: Uploading the file



Name
Password
Mail
Gender
DOB
Contact Number
State
Country
File Access Attribute
File Access Attribute

Fig-7: Data user registration

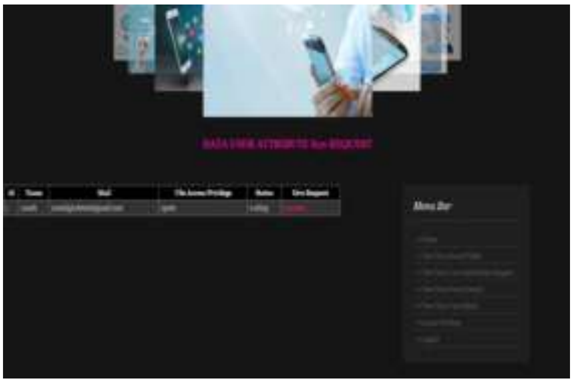


Attribute Key Request

Id	Name	Mail	Status	Give Request
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Fig-8: Attribute Key request

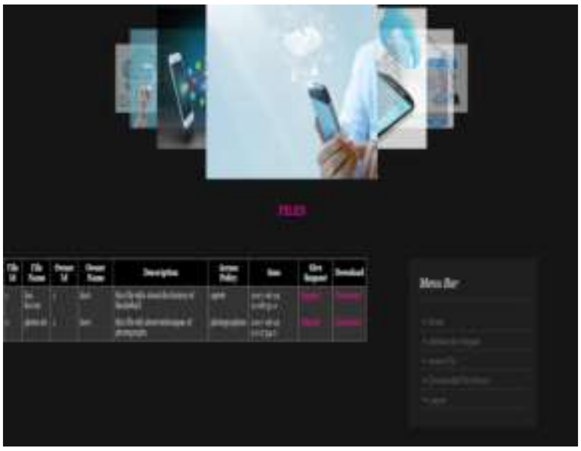
Figure-8 Data user on receiving UID sends attribute key request to the trusted authority. Figure-9 depicts generation of attribute key by TA which is sent to the data users mail ID



DATA USER ATTRIBUTE Key REQUEST

Id	Name	Mail	File Access Privilege	Status	Give Request
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Fig-9: Attribute key granted by trusted authority



FILES

File Id	File Name	Owner Id	Owner Name	Description	Access Policy	Date	Give Request	Download
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Fig-10: Request for Decrypt Key request

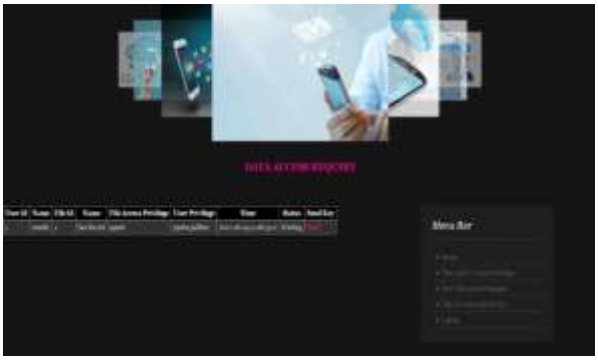


Fig-11: Data access request

Data user will be able to see the files uploaded by data owners and he will request decrypt key for the requested files Generation of decrypt key by cloud which is sent to the data users mail ID On receiving decrypt key, the data user will be able to download the file.

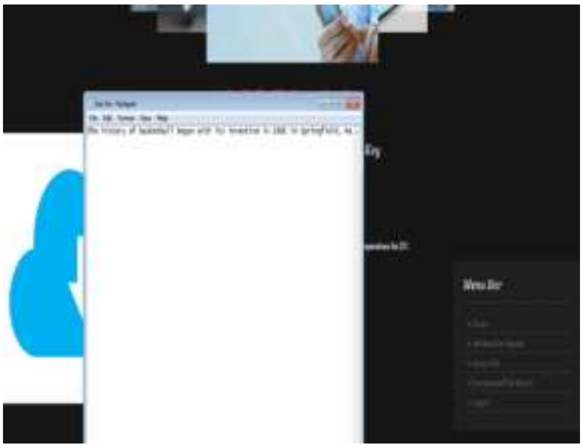


Fig-12: Downloaded files

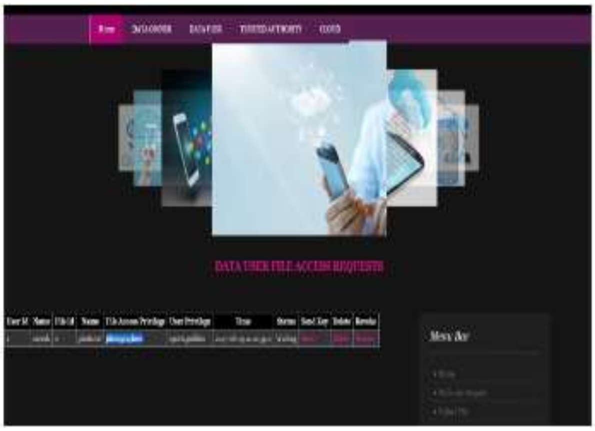


Fig-13: Revocation

Figure-13 depicts the request for decrypt key will be sent to data owner when the policy does not match. The data owner has the privilege to send revoke or delete the request.

CONCLUSIONS AND FUTURE WORK

The major objective behind the mobile cloud computing is to empower the mobile user by providing a seamless and rich functionality, regardless of resource limitation of mobile devices. Most of the techniques available are not appropriate for mobile cloud as they have exhaustive data processing. In this work we have presented a new scheme called Lightweight secure data sharing scheme to mitigate this problem. This scheme reduces exhaustive data processing and proxy servers are introduced for encryption and decryption services.

A provision is also provided for the data owners to send/revoke/delete data users request when the access policies do not match. In future work this work can be extended to solve security issues in the mobile cloud and new approaches to ensure data integrity can be employed. We can also study how to do cipher text retrieval over existing data sharing schemes.

REFERENCES

1. Bethencourt, J., Sahai, A., & Waters, B. (2007). Ciphertext-policy attribute-based encryption. In *Security and Privacy, 2007. SP'07. IEEE Symposium on* (321-334). IEEE.
2. Brakerski, Z. (2011). Efficient Fully Homomorphic Encryption from (Standard) LWE, FOCS.
3. Wang, Q., & Jin, H. (2011, June). Data leakage mitigation for discretionary access control in collaboration clouds. In *Proceedings of the 16th ACM symposium on Access control models and technologies* (103-112). ACM.
4. Skillen, A., & Mannan, M. (2013). On implementing deniable storage encryption for mobile devices.
5. Wang, W., Li, Z., Owens, R., & Bhargava, B. (2009, November). Secure and efficient access to outsourced data. In *Proceedings of the 2009 ACM workshop on Cloud computing security*, 55-66. Acm.
6. Maheshwari, U., Vingralek, R., & Shapiro, W. (2000, October). How to build a trusted database system on untrusted storage. In *Proceedings of the 4th conference on Symposium on Operating System Design & Implementation-4* (10). USENIX Association.
7. Yang, K., Jia, X., & Ren, K. (2013, May). Attribute-based fine-grained access control with efficient revocation in cloud storage systems. In *Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security*, 523-528. ACM.